対話型データ処理その－1－

文字列を使ったコードによるデータの選択

吉井守正（鉱床部）
Morimasa Yoshii

1 はじめに

筆者の所属する鉱床部にある横河ヒューレット・パッカード社製20型（YHP-9820A）卓上電子計算機は　手頃な会話型機として　多くの人に利用されてきた．　この計算機用のプログラムについては　数年前に　本誌“電卓シリーズ”（第275，276，277，282号）の中で紹介したのでご記憶の方もあろうかと思う．

さて　地質調査所が筑波学園都市に移転した機会に9820Aの上級新型機であるYHP-9845Tが導入された．この計算機は　やはり卓上型ながら　本格的な会話型計算機としての機能を備えている．　すなわち　同社固有の拡張BASICを言語とし　約180キロバイトのメモリー規模をもち　文字と図形が画面に写し出される陰極線管（Cathode Ray Tube）感熱式プリンタ　2台のカートリッジテープレコーダーを　標準的に装備している．　さらにプロッタやディジタイザとの接続により　多方面の応用が期待されて　楽しみな計算機である（第1図）．

そこで筆者は早速　これまで9820A用に作ってきたプログラムの中から　とくに人気の高かった主として岩石用化学分析データの処理プログラムを　9845T用に書き改めることにした．　その基本的な部分について紹介しよう．

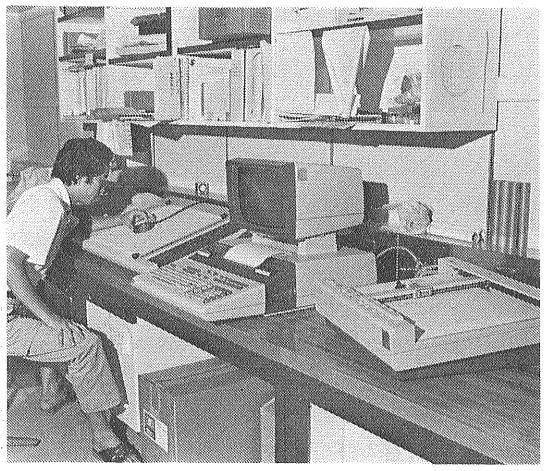
2 コードの設計

われわれが行うデータ処理では　計算機のメモリーに収められた多量なデータの中から　必要なものだけを任意に呼び出して処理するという場合が　一般的である．これを確実で能率的に行うやり方として　データに分類コードを付けておき　そのコードに従ってデータを選択する方法が9820A用プログラムで使われ　好評だった．これについては本誌第282号でも紹介した．　9845T用プログラムでも　この方式を引き継ぐことにした．

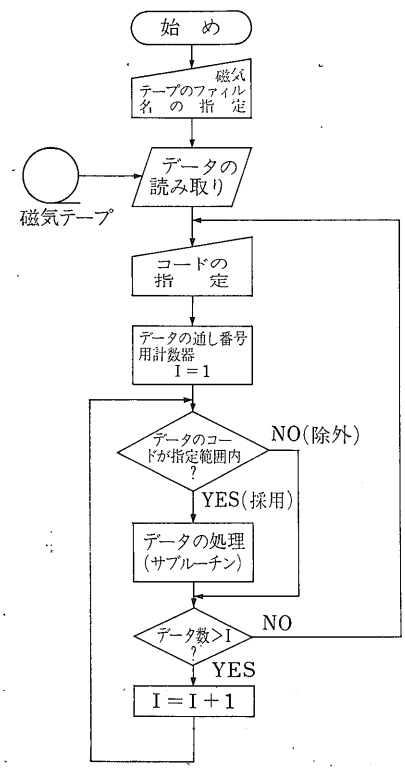
9845Tでは　数値のほかに文字もデータとして扱えるいわゆる文字列機能をもっているので　コードにもこれを採用した．　今回はその改良点を中心に述べよう．

データを処理する操作のあらましを第2図に示す．

データに付けるコードは　目下のところ　10文字の文字列から構成されており　これをつぎに述べる4つのサブコードに分けて使うのを標準としている．　つまり第1文字を第1サブコード　第2文字目以下は　3字ずつ



第1図　会話型の新鋭卓上電子計算機　YHP-9845T



第2図
データ処理プログラムのあらまし

このプログラムには終りがない．カートリッジテープの容量が計算機の容量とほぼ同じなので目下のところは　全部のデータを一度に計算機に読み込むようにしてある．

第1表　ASCII文字コードのおもなもの

文字	コード	文字	コード	文字	コード	文字	コード
・	46	F	70	V	86	l	108
0	48	G	71	W	87	m	109
1	49	H	72	X	88	n	110
2	50	I	73	Y	89	o	111
3	51	J	74	Z	90	p	112
4	52	K	75	a	97	q	113
5	53	L	76	b	98	r	114
6	54	M	77	c	99	s	115
7	55	N	78	d	100	t	116
8	56	O	79	e	101	u	117
9	57	P	80	f	102	v	118
A	65	Q	81	g	103	w	119
B	66	R	82	h	104	x	120
C	67	S	83	i	105	y	121
D	68	T	84	j	106	z	122
E	69	U	85	k	107	～	126

筆者のシステムで使う文字だけを掲げた．　点線で区切られた文字の間には記号がはさまれている．　コード番号は10進数表示のものだけを示した．

第2表　著者が試作した地質時代コード

コード	地質時代
ACA	Cambrian
DOR	Ordovician
GSI	Silurian
JDE	Devonian
MCB	Carboniferous
PPE	Permian
STR	Triassic
VJU	Jurassic
YCR	Cretaceous
bPA	Palaeocene
eEO	Eocene
hOL	Oligocene
kMI	Miocene
nPO	Pliocene
qPS	Pleistocene
tHO	Holocene

コードの第1字目がASCIIコードの順に配列されている．（ただし2字飛び）あとの2文字がその時代名の略号と読める．

3つに区切って　それぞれを第2　第3　第4　サブコードと呼ぶことにする．　その1例を示す．

コ　ー　ド　3 | ABC | 4GH | b5M

サブコード　1　2　3　4

これに使われるのは　数字（0～9）アルファベットの大文字（A～Z）と小文字（a～z）である．　原理的にはこのほかに記号も使えるのだが　記号の中には計算機に対するコマンドになっているものや　筆者の方式で特別な意味をもつものがあり　一般には使用を避けていただきたい．

4つのサブコードは　データの性質とにらみ合わせて任意に分類の定義をしてよい．　たとえば岩石試料などの場合は　産地　地質時代　岩質などというような分類項目が考えられる．　この場合　たとえば地質時代なら古いものから新しい方へ　火成岩の岩質なら苦鉄質から珪長質へといった具合に　一定の順序に従ってコードが配列されるようにするのが　ポイントである．

文字の序列は　計算機の場合　いわゆるASCIIコード(American Standard Code for Information Interchange)によって定められている．　そのおもなものを第1表に示す．　文字がASCIIコードの上で値をもっているわけだから　文字や文字の組同士が“大小関係”をもつと考えてよい．　筆者の作った方式では“．”を最少値“～”を最大値として　ASCIIコードの値が　これらの間にはさまる文字を使う．　すなわち．

・＜0＜……＜9＜A＜……＜Z＜a＜---＜z＜～

という関係になる．　もしサブコードの一部だけ使うようなときは　右寄りのサブコードだけを使うことにして使わない文字は“・”で埋めておく．　つまりつぎのように

・|...|KH.|.5M

として　字数だけは10文字にしておく．　空きの部分にスペースを使いたいところだが　あとで4つのサブコードに分解する行程で支障があるので使えない（こまかい説明は省略する）．

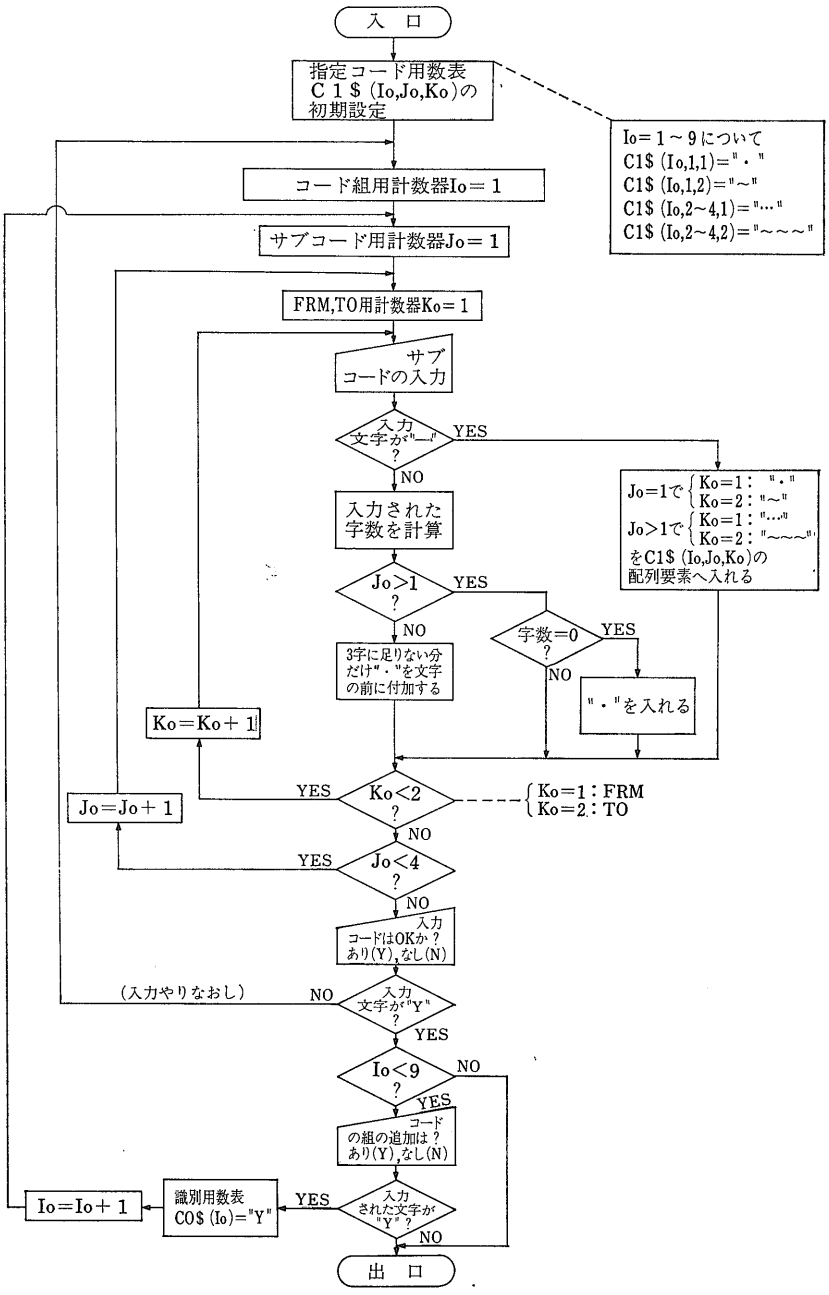
これら10個の数字と　52個のアルファベットを使うならば　コードの1文字について62通り　したがって3文字から構成されるサブコードの種類は　じつに

$62^3=238328$

もの種類に及ぶ．　つまり24万通り近い分類が　3文字で可能となる．

一般には　ひとつの事項をこれほどまでに区分する必要は起きないであろう．　そこでこの3文字を少しぜいたくに使い　その代りにサブコードを略号としても読み取れるように工夫すると便利である．

その考えを入れて　地質時代についてのサブコードを試作したので第2表に示す．　最初の文字のASCIIコー



第3図　コード指定の行程
使用者が指定したコードは三次元配列 C1$ (Io, Jo, Ko) に収められる．　一度指定したサブコードを取り消してそのサブコードを無視したいときは“－”を入れる．
入力するサブコードの字数が　規定の数に足らないときは　計算機が文字を右つめにして　空き間に“・”を打つ．

さらに第1字目は　アルファベットが2字飛びにしてある．　これは　あとから項目の追加ができるように　“味”を残したものである．　このようにサブコードを定義するときは　文字を少し飛ばして　余裕をもたせておき　将来起きるかも知れない　追加や変更に備えるのも設計のコツである．

これまでの実例からしても　コードの良し悪しがあとからのデータ処理の能率に大きく影響することがある．　だから少し時間をかけてでも　データの内容をよく分析して　周到なコードの設計をすることをおすすめする．

3　コードの指定

コード付きのデータは　入力用のプログラムによって計算機のメモリーへ入れられ　さらにテープのファイルにレコードされて保存される．　これらの行程については改めて執筆しよう．　ここでは　必要とするデータを　コードによって呼び出す操作について説明しよう．

データを処理するプログラムにはデータのコードを指定する行程があり　使用者は　4つのサブコードを入力する．　その際に各サブコードとも任意の範囲を指定できる．　計算機がCRT の画面を通じて　質問して来るので“FRM” (from) でASCII コード上　小さい値となる方のサブコードを入力し“TO”では大きい方のサブコードを入れる．

さきほどの地質時代の例では　たとえば三畳紀から白

ドの値が下へ向って大きくなり　そのサブコード全体のASCII コードの値も　下の方に大きくなる．　これで計算機用の序列が定まる．　そして2字目からあとの2文字は　時代名の略号としても読める仕組みになっている．

```
1000  ! ************************************
1010 Icodi: FOR I0=1 TO 9
1020  C1$(I0,1,1)="."
1030  C1$(I0,1,2)="~"
1040  NEXT I0
1050  FOR I0=1 TO 9
1060    FOR J0=2 TO 4
1070    C1$(I0,J0,1)="..."
1080    C1$(I0,J0,2)="~~~"
1090    NEXT J0
1100  NEXT I0
1110  RETURN
1120  ! ************************************
1130 Icod: !
1140  IMAGE "  Sub-Code:"2X,D,17X,D,15X,D,15X,D
1150  IMAGE 10X,"FRM"2X"TO"10X"FRM"3X"TO"8X"FRM"3X"TO"8X"FRM"3X"TO"
1160  IMAGE #"  Pair",2D,3X
1170  IMAGE #,3A,2X
1180  IMAGE #,6X
1190  PRINT USING 1140;1,2,3,4
1200  PRINT USING 1150
1210  FOR I0=1 TO 9
1220  PRINT USING 1160;I0
1230    FOR J0=1 TO 4
1240      FOR K0=1 TO 2
1250 Ipc: IF (J0=1) AND (K0=1) THEN INPUT ".  [CODE From (#1)]",C1$(I0,1,1)
1260      IF (J0>1) AND (K0=1) THEN INPUT "... [CODE From (#3)]",C1$(I0,J0,1)
1270      IF C1$(I0,J0,1)="-" THEN 1310
1280      IF (J0=1) AND (K0=2) THEN INPUT ".  [To CODE (#1)]",C1$(I0,1,2)
1290      IF (J0>1) AND (K0=2) THEN INPUT "...  [To CODE (#3)]",C1$(I0,J0,2)
1300      GOTO 1360
1310      IF J0=1 THEN C1$(I0,1,1)="."
1320      IF J0=1 THEN C1$(I0,1,2)="~"
1330      IF J0>1 THEN C1$(I0,J0,1)="..."
1340      IF J0>1 THEN C1$(I0,J0,2)="~~~"
1350      GOTO Ipc2
1360      L=LEN(C1$(I0,J0,K0))
1370      IF (J0=1) AND (L<=1) THEN Ipc2
1380      IF (J0>1) AND (L<=3) THEN 1410
1390      BEEP
1400      GOTO Ipc
1410      L=3-L
1420      C1$(I0,J0,K0)=RPT$(".",L)&C1$(I0,J0,K0)
1430 Ipc2: PRINT USING 1170;C1$(I0,J0,K0)
1440      NEXT K0
1450    PRINT USING 1180
1460    NEXT J0
1470    PRINT
1480    INPUT "CODE OKAY? (Y,N)",Z$
1490    IF Z$="Y" THEN 1540
1500    IF Z$="N" THEN Icod
1510    BEEP
1520    GOTO 1480
1530    IF I0=9 THEN 1580
1540    INPUT "INPUT ANOTHER PAIR OF CODES? (Y,N)",C0$(I0)
1550    IF C0$(I0)<>"Y" THEN 1580
1560    PRINT "--OR"
1570  NEXT I0
1580  RETURN
1590  ! ************************************
```

第4図　コード指定行程のプログラムリスト
この行程は　サブルーチンになっている.
ラベル Icodi は　指定コード用数表の初期設定の入口
ラベル Icod は　コード指定の入力行程の入口

亜紀までを指定するときには

```
FRM     TO
STR     YCR
```

という具合になる.　このようにして　第1サブコードから第4サブコードまでについて　それぞれ範囲を指定する.　各サブコードを入力した結果は　たとえばつぎのようにディスプレイされる.

```
   1        2        3        4
FRM TO FRM TO FRM TO FRM TO
 B   G STR YCR 4GH MN.  ..N .gE
```

上のような　第1サブコードから第4サブコードまでの組を“コードの組”と呼ぶことにしよう.　コードを指定する行程では　このコードの組を同時に9組まで入力できる.　つまり　ひとつのサブコードに対して飛び飛びの範囲が指定できる.　地質時代のサブコードの場合は　実際にこのような指定が行われるかどうか疑わしくて　適切な例とは言い難いが　単なる1例として掲げると

```
          FRM     TO
第 1 組   DOR     JDE
第 2 組   STR     YCR
第 3 組   hOL     kMI
```

といった指定が9組までできる.

処理のプログラムは第2図で見るとおり　終りがなく全部の処理が完了すると　コード指定の行程へ再び戻る.　したがって　いま述べたコードの組は　処理を反復すれば　理論的には　無限に追加指定できることになる.

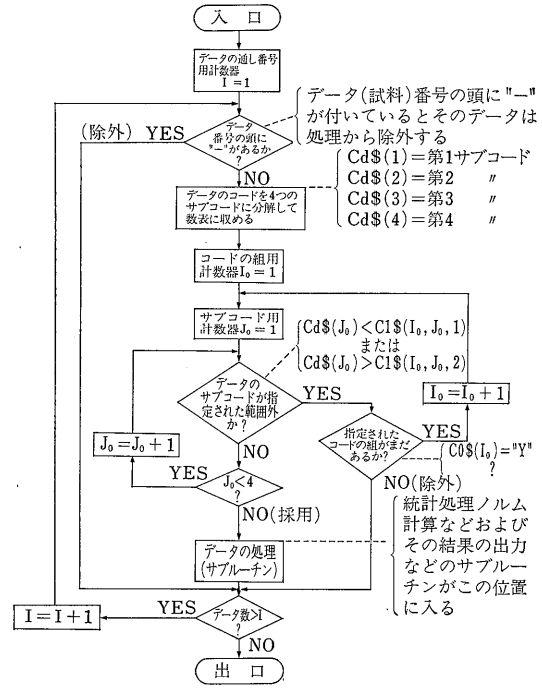
もし　データを　サブコードに関係なく呼び出したいときは　第1サブコードでは　第2サブコードから第4サブコードでは

```
FRM     TO
...     ~~~
```

と指定する.　これでそのサブコードの全範囲を指定したことになる.

このような使用例は実際には多いので　メモリーの数表にあらかじめ“·(···)”や“~(~~~)”を入れておき　使用者が　文字を入れないで　プログラムステップを進めるキイを押すと　これらの記号が　そのまま数表

```
FRM     TO
·       ~
```



第5図　コードによるデータ選択の行程
データに付いているサブコードは Cd-$（1～4）に　使用者が指定したサブコードは C1$ (I₀, J₀, 1～2) に入っている.　指定されたコードの組が続くときには C0$ (I₀) に“Y”が入れられる.

```
1840  FOR I=1 TO Sc(1)
1850  IF No$(I,1)[1,1]="-" THEN Ad
1860  Cd$(1)=No$(I,2)[1,1]
1870  Cd$(2)=No$(I,2)[2,4]
1880  Cd$(3)=No$(I,2)[5,7]
1890  Cd$(4)=No$(I,2)[8]
1900  FOR I0=1 TO 9
1910    FOR J0=1 TO 4
1920  IF (Cd$(J0)<C1$(I0,J0,1)) OR (Cd$(J0)>C1$(I0,J0,2)) THEN 1950
1930    NEXT J0
1940    GOTO Pick
1950    IF C0$(I0)<>"Y" THEN Ad
1960  NEXT I0
1970 Pick: GOSUB Nor6
1980  GOSUB Plot
1990 Ad: NEXT I
```

第6図　データ選択行程のプログラムリスト
"Pick" というラベルの付いたステップが　狭義のデータ処理サブルーチンへの出口

に残る仕組にしてある.

コード指定の行程の流れ図を第3図に　そのプログラムリストを第4図に示す.

4　データ選択の行程

指定されたサブコードは（9，4，2）の規模をもつメモリーの数表C1$ (I_0, J_0, K_0) に入れられる（第4図）. この数表の　配列の要素は　I_0がコードの組番号　J_0がサブコードの番号　K_0は1がFRM　2がTOに対応する.

一方　データに付いているコードは　4つのサブコードに分解される.　その上で　そのサブコードが指定された範囲内にあるかどうか　判断される.　判断は当然ASCII コードとしての大小関係による.　この行程の流れ図を第5図に　そのプログラムリストを第6図に示す.

10文字で構成されているコードから　4つのサブコードを割り出すには　文字列から副文字列を得る機能を用いる.　データのコードは　No$（I，2）という数表に入れられている.　コードの10文字を第1サブコードから第4サブコードに分けて　それぞれを　Cd$($J_0$) という数表に収める操作のプログラムは　第6図にもあるとおり

```
Cd$(1)=No$(I,2)[1,1]
Cd$(2)=No$(I,2)[2,4]
Cd$(3)=No$(I,2)[5,7]
Cd$(4)=No$(I,2)[8]
```

となる.　こうして4つの数表に収められた　ひとつのデータがもっているサブコードと　使用者に指定されてC1$ (I_0, J_0, K_0) に入いっているサブコードとを　ひとつひとつ比較する.　これらの行程は　筆者の作ったシステムの心臓部とも言える部分である.

データの選択に対して　サブコードの指定範囲（FRM, TO）は　論理上はorで結ばれ　4つのサブコードは互いに and の結合になっている.　またコードの組はorで連結されている.　このように　この行程は一見複雑そうに思えるが　実際には　流れ図や　プログラムリストで見るとおり　I_0 と J_0 の2個の計数器を回して　ループ処理しているだけである.

5　データ処理のサブルーチン

使用者のコード指定と合致して　選び出されたデータは処理行程へ移る.　9845T用のプログラムではこの処理行程は　一般に　選択行程のサブルーチンになっており　主プログラムから分岐して　選択されたデータについての一連の処理（計算処理や図示などの出力）をして再び主プログラムへ戻る.　ノルム計算を伴うプログラムの場合でも　ノルム計算自身は　データ処理の一部に過ぎないので　同じようにサブルーチンになっている.

筆者のプログラムでは　これら狭義のデータ処理行程をサブルーチンとして　主プログラムに付加しているだけなので　この部分を入れ換えることにより　種々の目的の処理ができる.　そして　コードの指定など　使用者がキイボードから行なう操作の　主要な手順が　つねに一定になるので　それだけ使いやすいことにもなる.